IN THE CLAIMS:

1. (Currently Amended) A computer method for comparing a data set of a source storage system replicated on a destination storage system~~a first directory comprising unique elements with a second directory comprising unique elements~~, comprising:

(a) hashing~~for~~ entries~~each entry~~ in ~~the~~ a first directory and~~,~~ placing ~~a~~ hash values for the entries ~~of the entry~~ in a hash table, the first directory stored on ~~a~~ the source storage system;

(b) selecting an entry ~~of~~ in ~~the~~ a second directory and hashing the selected entry to provide a selected hash value, the second directory stored on ~~a~~ the destination storage system, wherein the source storage system and the destination storage system are operatively connected via a network;

(c) looking up a match between the selected~~a~~ hash value of the selected entry and a first~~the~~ hash value ~~of the entry~~ in the hash table;

(d) removing the first hash value from the hash table~~,~~ in response to the match between the selected hash value of the selected entry and the first hash value ~~of the entry~~ in the hash table~~, the hash value of the entry from the hash table~~;

(e) determining if an additional entry in the second directory ~~entry~~ exists;

(f) looping to step (b) in response to identifying the additional entry in the second directory ~~entry~~; and

(g) reporting a difference between the first directory and the second directory in response to at least one hash value entry remaining in the hash table.

2. (Currently Amended) The method of claim 1 further comprising identifying that the selected entry is second directory unique~~,~~ in response to ~~not locating the~~no match between the selected hash value of the selected entry and the first hash value ~~of the entry~~ in the hash table,~~ that the hash value of the selected entry is second directory unique~~.

3. (Currently Amended) The method of claim 1 further comprising performing a remedial function~~,~~ in response to ~~not locating the~~no match between the selected hash value of the selected entry and the first hash value ~~of the entry~~ in the hash table~~, a remedial function~~.

4. (Currently Amended) The method of claim 3 wherein the remedial function comprises deleting the selected entry ~~of~~ in the second directory.

5. (Currently Amended) The method of claim 1 further comprising identifying any remaining hash value entry in the hash table as being first directory unique in response to determining that no additional entry in the second directory exists~~existing, any remaining hash value entry in the hash table as being first directory unique~~.

6. (Currently Amended) The method of claim 1 further comprising performing a remedial function in response to determining that no additional entry in the second directory exists~~existing, a remedial function~~.

7. (Currently Amended) The method of claim 6 wherein the remedial function comprises deleting the selected entry in~~of~~ the first directory.

8. (Previously Presented) The method of claim 6 wherein the remedial function comprises transferring the selected entry from the first directory to the second directory.

9. (Previously Presented) The method of claim 1 wherein the data are organized by a RAID system.

10. (Original) The method of claim 1 wherein the hash table comprises a B-tree.

11. (Original) The method of claim 1 wherein the hash table comprises a fast lookup data structure.

12. – 16. (Cancelled)

17. (Currently Amended) A computer system for comparing a first data set with a second data set, comprising:

(a) means for hashing entries in the first data set and means for placing ~~a~~ hash values for the entries~~of each entry of the first data set~~ in a hash table, the first data set stored on a source storage system;

(b) means for selecting an entry ~~of~~ in the second data set and hashing the selected entry to provide a selected hash value, the second data set stored on a destination storage system, wherein the source storage system and the destination storage system are operatively connected via a network;

(c) means for looking up a match between the selected~~a~~ hash value of the selected entry and a first~~the~~ hash value ~~of the entry~~ in the hash table;

(d) means for removing the first hash value from the hash table~~,~~ in response to the match between the selected hash value of the selected entry and the first hash value ~~of the entry~~ in the hash table~~, the hash value of the entry from the hash table~~;

(e) means for determining if an additional entry in the second data set ~~entry~~ exists;

(f) means for looping to step (b) in response to identifying the additional entry in the second data set ~~entry~~; and

(g) means for reporting a difference between the first data set and the second data set in response to at least one hash value entry remaining in the hash table.

18. (Original) The system of claim 17 wherein the hash table comprises a B-tree.

19. (Currently Amended) A computer readable storage medium containing executable program instructions executed by a processor, comprising:

(a) program instructions that hash entries~~for each entry~~ in a first data set and~~,~~ program instructions that place ~~a~~ hash values for the entries ~~of the entry~~ in a hash table, the first data set stored on a source storage system;

(b) program instructions that select an entry ~~of~~ in ~~the~~ a second data set and program instructions that hash the selected entry to provide a selected hash value, the second data set stored on a destination storage system, wherein the source storage system and the destination storage system are operatively connected via a network;

(c) program instructions that look up a match between the selected~~a~~ hash value of the selected entry and ~~the~~ a first hash value ~~of the entry~~ in the hash table;

(d) program instructions that remove the first hash value from the hash table~~,~~ in response to the match between the selected hash value of the selected entry and the first hash value ~~of the entry~~ in the hash table~~, the hash value of the entry from the hash table~~;

(e) program instructions that determine if an additional entry in the second data set ~~entry~~ exists;

(f) program instructions that loop to step (b) in response to identifying the additional second data set entry; and

(g) program instructions that report a difference between the first data set and the second data set in response to at least one hash value entry remaining in the hash table.

20. (Currently Amended) A computer method for comparing a first data set with a second data set, comprising:

creating a hash table of hashed value entries from entries of the first data set, the first data set stored on a source storage system;

locating an entry from the second data set and creating a hash value entry ~~from the second data set~~ in the hash table, the second data set stored on a destination storage system, wherein the source storage system and the destination storage system are operatively connected via a network;

removing, in response to locating the hash value entry from the second data set in the hash table, the hash value entry from the hash table; and

recording, in response to at least one hash value entry remaining in the hash table, a difference between the first data set and the second data set.

21. (Currently Amended) A computer method for comparing a first data set with a second data set, comprising:

creating a hash table of hash value entries of the first data set, the first data set stored on a source storage system, the hash table comprising one or more hashed values of the first data set;

determining whether a hashed value of an entry of the second data set is identical to a first hashed value entry in the hash table, the second data set stored on a destination storage system, wherein the source storage system and the destination storage system are operatively connected via a network;

removing, in response to determining that the hashed value of the second data set is identical to the first hashed value entry in the hash table, the first~~identical~~ hashed value entry from the hash table;

recording, in response to determining that the hashed value of the second data set is not identical to the first hashed value entry in the hash table, the hashed value of the second data set as unique; and

reporting a difference between the first data set and the second data set in response to at least one hashed value entry remaining in the hash table.

22. (Currently Amended) A computer method for comparing a first data set with a second data set, comprising:

(a) selecting an entry ~~from~~ of the first data set;

(b) determining if a hashed value of the selected entry of the first data set is in a hash table, the hash table comprising one or more hashed values of the first data set;

(c) adding the hashed value of the selected entry of the first data set to the hash table~~,~~ in response to determining that the hashed value of the selected entry of first data set is not in the hash table~~, the hashed value of the selected entry of the first data set to the hash table~~;

(d) removing from the hash table the hashed value of the selected entry of the first data set~~,~~ in response to determining that the hashed value of the selected entry of the first data set is in the hash table~~, the hashed value of the selected entry of the first data set~~;

(e) selecting an entry ~~from~~ of the second data set;

(f) determining if a hashed value of the selected entry of the second data set is in the hash table, the hash table further comprising one or more hashed entries of the second data set;

(g) adding the hashed value of the selected entry of the second data set to the hash table~~,~~ in response to determining that the hashed value of the selected entry of the second data set is not in the hash table~~, the hashed value of the selected entry of the second data set to the hash table~~;

(h) removing from the hash table the hashed value of the selected entry of the second data set~~,~~ in response to determining that the hashed value of the selected entry of the second data set is in the hash table~~, the hashed value of the selected entry of the second data set~~;

(i) continuing (a) through (d) and (e) through (h) respectively for all entries in the first and the second data sets until both the first and the second data sets have been completely processed; and

(j) reporting a difference between the first data set and the second data set in response to at least one hashed value remaining in the hash table.

23. (Currently Amended) The method of claim 22 wherein ~~the~~ adding the hashed value of the selected entry of the first data set to the hash table further comprises adding information ~~with the hashed value of the selected entry of the first data set~~ identifying the hashed value of the selected entry of the first data set as originating from the first data set.

24. (Currently Amended) The method of claim 22 wherein adding the hashed value of the selected entry of the second data set to the hash table further comprises adding information ~~with the hashed value of the selected entry of the second data set~~ identifying the hashed value of the selected entry of the second data set as originating from the second data set.

25. (Previously Presented) The method of claim 22 wherein the data are organized by a RAID system.

26. (Previously Presented) The method of claim 22 further comprising:

(k) recording all hashed value entries remaining in the hash table as being unique to either the first data set or the second data set.

27. (Original) The method of claim 22 wherein the hash table comprises a B-tree.

28. (Original) The method of claim 22 wherein the hash table comprises a fast lookup data structure.

29. (Original) The method of claim 22 wherein the first data set comprises a set of directory entries on a source system.

30. (Original) The method of claim 22 wherein the second data set comprises a set of directory entries on a destination system.

31. (Original) The method of claim 22 wherein the first data set and second data set are on different storage devices.

32. (Currently Amended) A system for performing a consistency check of a source directory replicated to a destination directory by comparing entries in the source and destination directories, comprising:

one or more storage devices operatively connected to a computer configured to store one or more entries of a group consisting of the source directory and the destination directory; and

a processor of the computer configured to execute a process, the process configured to compare entries ~~in~~ of the source directory with entries ~~in~~ of the destination directory by storing a hash value of each entry of the source directory and the destination directory in a hash table, the process further configured to remove from the hash table any hash value of the destination directory which matches any hash value of the source directory~~ and the destination directory~~.

33. (Original) The system of claim 32 wherein the process executes on a computer associated with the source directory.

34. (Original) The system of claim 32 wherein the process executes on a computer associated with the destination directory.

35. (Cancelled)

36. (Currently Amended) A system for performing a consistency check of a source directory and a destination directory, comprising:

one or more storage devices operatively connected to a computer configured to store one or more entries of a group comprising ~~of~~ the source directory and the destination directory; and

a processor of the computer configured to execute a process, the process configured to compare entries ~~in~~ of the source directory with entries of~~in~~ the destination directory by storing a hash value of each entry of the source directory in a hash table, the process further configured to remove any entry from the hash table which matches any hash value of the destination directory.

37. (Currently Amended) A computer readable storage medium containing executable program instructions executed by a processor, comprising:

(a) program instructions that select an entry ~~from~~ of a first data set, the first data set stored on a source storage system;

(b) program instructions that determine if a hashed value of the selected entry of the first data set is in a hash table, the hash table comprising one or more hashed values of the first data set;

(c) program instructions that add the hashed value of the selected entry of the first data set to the hash table~~,~~ in response to determining that the hashed value of the selected entry of first data set is not in the hash table~~, the hashed value of the selected entry of the first data set to the hash table~~;

(d) program instructions that remove from the hash table the hashed value of the selected entry of the first data set~~,~~ in response to determining that the hashed value of the selected entry of the first data set is in the hash table~~, the hashed value of the selected entry of the first data set~~;

(e) program instructions that select an entry ~~from~~ of a second data set, the second data set stored on a destination storage system;

(f) program instructions that determine if a hashed value of the selected entry of the second data set is in the hash table, the hash table further comprising one or more hashed entries of the second data set;

(g) program instructions that add the hashed value of the selected entry of the second data set to the hash table~~,~~ in response to determining that the hashed value of the selected entry of the second data set is not in the hash table~~, the hashed value of the selected entry of the second data set to the hash table~~;

(h) program instructions that remove from the hash table the hashed value of the selected entry of the second data set~~,~~ in response to determining that the hashed value of the selected entry of the second data set is in the hash table~~, the hashed value of the selected entry of the second data set~~;

(i) program instructions that continue (a) through (d) and (e) through (h) respectively for all entries in the first and the second data sets until both the first and the second data sets have been completely processed; and

(j) program instructions that report a difference between the first data set and the second data set in response to at least one hashed value remaining in the hash table.

38. (Previously Presented) The computer readable medium of claim 37 further comprising program instructions that alternate in selecting entries from the source and destination directories.

39. (Currently Amended) The method of claim 1 wherein reporting comprises recording the difference on a storage device operatively connected to the source storage system.

40. (Currently Amended) The method of claim 22 wherein reporting comprises recording the difference on a storage device operatively connected to a computer.

41. (Previously Presented) The system of claim 32 wherein the process is further configured to add to the hash table any hash value which does not match any hash value of the source directory and the destination directory.